Principles of Information Security project specification

Freely distributable tools for finding web vulnerabilities

Tomáš Meravý Murárik

1. Theoretical Part

The theoretical part of this project will consist of three main sections.

1.1. Introduction

This section will familiarize the reader with the problems this project aims to tackle. As web applications comprise many components, a thorough explanation will be provided to ensure the reader has the necessary comprehension.

1.2. CWE Classification

This section will provide a brief explanation of Common Weakness Enumerations (CWEs) and their classification. Given the vast number of existing CWEs, only a select few will be chosen to measure the performance of the tools in this project. The intentional implementation of these specific CWEs will serve as a key point of comparison between the selected security tools. The classification will be based on the NIST National Vulnerability Database.

1.3. List and Characterization

This section will provide a concise overview and characterization of the specific vulnerabilities on which this project will focus.

2. Purpose of Project

This project will compare different open-source tools for scanning servers and finding vulnerabilities in web applications.

These applications are highly varied, with most being multi-functional black-box tools, which are capable of detecting SQL injections, subdomain guessing, file detection, credential guessing, and more. One exception is Dirsearch, which is application built and optimized for a specific task. It is a program designed solely for blindly discovering hidden files and directories using a predefined dictionary.

3. Tools and Tests

The tools for this project were selected based on their differing implementations and approaches to detecting the CWEs being tested. The tools used will include Dirsearch, w3af, Nikto, Wapiti, Vega, and Grabber. Applications used as targets will be PHP web apps with varying implementations. Tested will be simple web apps with few lines of php all the way to larger projects written in laravel framework as well as DVWA.

4. Methodology

This section will explain the systematic approach which testing will take to ensure fair and consistent comparison.

4.1. Test Environment

All tests will be conducted within a controlled environment to prevent any unintended network interference. Since tests will be on web application and it's functionality, tests will be done on a localhost server as this is easiest option which does not impact performance in any way or results.

4.2. Testing Procedure

To ensure consistent comparison, all tests will follow the following procedure.

1. **Tool configuration** Each tool will be run with default configuration against target to simulate out-of-the-box experience. Tool will later be executed with optimal configuration for best output.
2. **Data collection** All data the tool output will be logged including false positives, false negatives, and time it took for tool to complete testing.

4.3. Tests

- Hidden File Discovery

Nikto, a well-known software for testing various web application vulnerabilities, will be tested against Dirsearch, a more specialized program designed solely for crawling target websites to find hidden files and directories. This functionality will also be tested in w3af, Wapiti, Vega, and Grabber.

The tests for hidden file scanning will be designed to evaluate every aspect of this capability. Examples include finding files with simple names (`db.log`), complex names (`xd320812.pdf`), and files referenced within JavaScript files.

- SQL Injection Detection

SQL injection is an area where many security tools excel. As one of the most powerful web application attacks, it can range from simple, visible cases to highly complex ones. Nikto, w3af, Wapiti, Vega, and Grabber will be tested on their ability to detect these vulnerabilities, from basic concatenation-based SQL injections to more advanced cases.

- Outdated Component Reporting

The final comparison will focus on identifying outdated components on websites. Since Nikto, w3af, and Wapiti all feature this capability, their implementations will be tested and their performance compared.

5. Project Timeline

The project will be executed according to the following schedule(week numbering is based on my study program):

- **Week 3-4:** Environment Setup & Tool Familiarization
- **Week 3-4:** Test Case Development and CWE Implementation
- **Week 7-8:** Testing and Data Collection
- **Week 9-10:** Data Analysis and Visualization
- **Week 11:** Report Writing and Finalization

6. Resources

Resources which this project uses will be NIST documentation and classifation, Owasp, Laravel documentation, and many more, all of which will be cited in final report.

7. Expected Outcome

Results of this project will yield the following deliverables:

- Report which contains comparison of results acquired by testing. This report clearly detail methodology, results and analysis.
- Graphs and tables which visualize compared results and performance accross all tested categories.
- Raw data and configuration scripts used during testing.

8. Goal

The goal of this project is to compare and summarize the results from different open-source web vulnerability scanners. By systematically testing these tools, this project aims to identify clear differences in their performance and precision, ultimately determining the best tool for each tested category.

9. Conclusion

By rules of STU FIIT I am now disclosing that generative AI has been used in the text above for text formating.